# Improving Grid computing performance prediction using weighted templates

**Ariel Goyeneche**, Dr Gabor Terstyanszky, Dr Thierry Delaitre, Prof Stephen Winter

Centre for Parallel Computing, Cavendish School of Informatics
University of Westminster, 115 New Cavendish Street, London, W1W 6UW

## Abstract

Understanding the performance behavior of Grid components to predict future Job submissions is considered one of the answers to automatically select computational resources to match users' requirements maximizing its usability. Job characterization and similarity are key components in making a more accurate prediction. The purpose of this paper is to test how current data mining and statistical solutions that define jobs similarity perform in production Grid environments and to present a new method that defines template using two set of characteristics with different priorities and weights the templates prediction accuracy level for future use. The results show that the new method achieves in average a prediction errors than is 54 percent lower than those achieved by using dynamic templates.

## 1. Introduction

The increasing complexity of Grid systems presents a new challenge for Grid schedules: how computational resources can be automatically selected to match users' requirements maximising its usability.

Understanding the performance behaviour of Grid components to predict future job submissions is considered one of the answers for this challenge. The execution time prediction can be used to allocate resources in advance and inform before hand finishing time and running costs. Without this feature, the Grid users may only look forward to see their jobs completed at some point in the future without indication of Resource utilization.

Historical data mining and statistical methods have been used to tackle this problem. In an ideal scenario, the historical data would be collected by a centralized knowledge acquisition system that solves distributed and semantically heterogeneous problems. During the next stage the data would be mined in an inductive learning process in order to generate models of the Grid environment that not only holds the information and rules to forecast future job submissions, but also helps in the detection of performance anomalies. If possible, this process should be repeated and validated against real data in order to absorb Gird environments changes as well as minimize prediction errors.

In this process a key component in making a more accurate prediction is the characterization of jobs and the definition of similarity. The objective is to understand which set of characteristics is the best fit to group jobs together in order to use the clustered information to forecast future submissions.

In this paper an existing solution that defines similarity using dynamic templates tested with homogeneous workloads is implemented with the objective to examine its performance and accuracy in a heterogeneous production Grid environment. Based on this result, a new solution that takes into account Grid features is presented and validated.

Instead of treating all characteristics with the same level of influence, the new solution defines similarity using two set of characteristics: a high priority first set that concentrates the relevant Grid characteristics that bind and describe Jobs and Users and a second set where any other characteristic is placed. A function based on historical prediction accuracy weighs any possible template composes of characteristics from the first set. For all templates with the same prediction accuracy, the dynamic template solution is applied but this time also taking into account the second set of characteristics. This research shows that defining similarity using two set of characteristics in Gird environments achieves in average prediction errors than are 54 percent lower than those achieved by using dynamic templates.

The remainder of this paper is organized as follows: In Section 2 the relevant Grid job characterization approaches are presented and analyzed. Section 3 introduces the current research motivation and the challenges to be faced when job characterizations are implemented in production Grids. In Section 4 the target environment used to examine and evaluate the job characterization methods, the UK National Grid Service, is described together with the historical information. In Section 5 a new characterization method and prediction techniques are presented. They are evaluated using the NGS workload, and compared and validated against the dynamic template solution. The final sections of this paper contain the conclusion of this work and future challenges to be addressed.

## 2. Grid jobs characterization.

Current performance prediction solutions for high performance applications can be divided in three main groups: prediction techniques based on mathematical methods, prediction techniques based on simulations, and prediction techniques based on data mining or statistical methods.

Existing performance prediction solutions based on mathematical methods, such as the Gamma Model [1][2] or the usage of time series presented by Dinda in [3][4] or Yuanyuan [5], cannot be always applied to different type of data and usually require a precise normalization of the input variables for assuring the correctness of the resulting conclusions. Performance solutions based on simulators [6][7][8] usually work off-line, are relatively slow, and are often used as a complementary tool that needs to be verified against real application results. Also, such simulators do not consider the importance of models that dynamically reflect any change in a Grid environment.

In order to overcome these limitations data mining and statistical methods have been applied to historical information from job submissions and Resource utilization (workload logs) to understand and predict Grid components performance behaviors. A relevant example in this group was carried out by Warren Smith [9][10] who uses dynamic templates for identifying good patterns for a given workload and statistical estimators. A similar approach is presented in GPRES [11] model which groups analogous jobs based on static or dynamic templates coexisting at the same time. Hui Lui et al [12] presented a new prediction technique for wait queue time and runtime using the K-Nearest Neighbour technique, and Goyeneche et al [13] demonstrate how the use of data mining techniques applied to Grid computing workload logs can be adopted to create a model of the Grid environment, understand its components performance, and predict their future behaviors.

The key component of the data mining or statistical methods is the definition of job's similarity: the historical information of a job can be used to predict future *similar* jobs behaviour.

A relevant example of job characterization and similarity was done by Downey [14]. He used a technique that models malleable jobs (jobs that can change the number of processors on which they are executing at run time in response to an external command) using the average parallelism of a program and its variance in parallelism. His procedure characterizes all applications in the workload, then models the cumulative distribution functions of the runtime in each category, and finally uses these functions to predict application runtime. He presents a two part function based on the variance in the degree of parallelism. When the model fits observed speedup curves well he is able to summarize a job behaviour by saving only the average parallelism of a program and its variance in parallelism.

But Downey's definition of similarity uses very few characteristics that produce misjudgements. In order to improve this problem Gibbon [15] introduced the idea of templates compose of different characteristics to group similar jobs (See Table 1 and Gibbon's column in Table 2). Given that the number of templates is fixed, one of the main problems of Gibbon's solution is that too many unrelated jobs are grouped together. Gibbons himself suggested that one of his solution's improvements is to vary the templates in the experiments to determine the each parameter sensitive.

| Template | Predictor |
|---|---|
| (u, e, n, age) | Mean |
| (e, n , age) | Mean |
| (n, age) | Mean |
| (u, e) | Lineal regression |
| (e) | Lineal regression |

Table 1 Templates used by Gibbons

Therefore Smith [9][10] presented another solution based on Gibbson's idea using a non-fix set of templates. Smith uses two algorithms called Greedy and Genetic to dynamically define the set of templates to be used or forecasting jobs performance. Both algorithms

take as an input a workload with a number of characteristics (See Smith's column in Table 2) and produces as an output a template set. Each template set is used by another algorithm that after three different faces (initialization, prediction and incorporation of historical information) produces a set of run-time prediction and associated confidence intervals.

| Characteristic | Gibbons | Smith |
|---|---|---|
| User (**u**) | Y | Y |
| Executable (**e**) | Y | Y |
| Number of Nodes (**n**) | Y | Y |
| How long application was executed (**age**) | Y | Y |
| Type (Batch, Interactive Serial, Parallel) (**t**) | | Y |
| Queue (**q**) | | Y |
| Class (**c**) | | Y |
| Loadleveler Script (**s**) | | Y |
| Arguments (**p**) | | Y |
| Network Adaptor (**na**) | | Y |
| Submission Time (**ut**) | | Y |
| StartTime (**st**) | | Y |
| RunTime (**rt**) | | Y |

Table 2 Characteristic used in templates

In [10] Downey, Gibbson and Smith's techniques where applied to different workloads. This experiment showed that Smith's solutions performed between 4 to 46 percentages better than Gibbson's solution, because of using other jobs characteristics and dynamic templates. Also, it is shown that Smith's solutions performed between 27 to 60% better than Downey's best performance.

## 3. Research motivation.

The solutions described in Section 2 base their job's similarity definition on the idea of having a right number of characteristics that can describe and group together similar jobs. Among them, it was demonstrated that using a dynamic method to group those set of characteristics (templates) together with normal distribution methods for selecting the estimate improves the performance prediction.
But, in spite of being a practical idea tested with homogeneous workloads, the dynamic template method may not usually work in heterogeneous production Grid computing environments because of the following reasons:

- There are key characteristics that are not always normalized in Grid environments. For instance, Grid users not always provide a unique job name or identification across several Grid nodes.
- There are some other important characteristics that are hidden or not shown. A common Grid user routine is to include in the executable script the set of parameters; therefore they are hidden and not available for analysis. In that case, an input parameter change that may produce a different type of performance cannot be considered for future submissions.
- Even though if the identification of jobs and publication of parameters can be solved, the use of only the walltime mean with the smallest confident of all possible templates may produce the grouping of jobs that are not related to each other, rather than by this function, and therefore generate misjudgements in future predictions..
- And last but not least, it is not clear that job walltime historical information can usually be described with normal distributions.

Therefore, the motivation of this research is to assess how well the use of dynamic templates perform in a production Grid environment and whether a different method for selecting the best estimate can be applied to obtain a better result.

## 4. NGS Performance prediction using dynamic templates.

### 4.1 The NGS workload logs.

The National Grid Service (NGS) [16] funded by JISC, EPSRC, and CCLRC, was created in October 2003. The service entered full production in September 2004 and provides a Grid based infrastructure to support UK e-scientists' research. NGS offers a range of computational and data resources, which can be accessed remotely using single sign-on mechanisms based on digital certificates.
The NGS is built around four core nodes and a growing list of NGS partner sites that support a diverse range of software applications. Some of them are available across all clusters, while some are only available on one or more clusters.
All NGS resources run compatible Grid middleware based on the Globus Toolkit 2 [17] for the execution of computational tasks, data

transfer and resource discovery. Additional middleware for data management is provided on the NGS core nodes in the form of the Storage Resource Broker and OGSA-DAI. At the time of colleting the workload data all four core nodes were running PBS Pro [18] as a cluster job Manager spawned by the Globus Gatekeeper to submit (query, report, send results back) jobs on the compute nodes.
With the aim to investigating the Grid job characterization, from July 2006 to October 2006 (see Table 3), a detailed level of raw data has been collected from the four NGS core node sites containing a record of all job submission information. The original logs that came in different formats were filtered and processed enabling the cleaning of abnormal events and leaving only the sections relevant for the analysis.

| Grid Node | From Date | To Date | Entries |
|---|---|---|---|
| grid-compute.leeds.ac.uk | 10/07/2006 16:05 | 19/10/2006 13:24 | 99263 |
| grid-compute.oesc.ox.ac.uk | 10/07/2006 15:54 | 19/10/2006 13:00 | 767630 |
| grid-data.man.ac.uk | 10/07/2006 16:10 | 10/10/2006 15:10 | 66747 |
| grid-data.rl.ac.uk | 10/07/2006 16:10 | 19/10/2006 13:22 | 122135 |

Table 3 NGS logs dates and entries

The most significant NGS fields for this study are:

- recorddate: Date and time when the information was extracted
- gridnode (gn): Name of the NGS Grid node
- jobId: Unique identification of the job in a Grid Node
- jobState: The state of the job
- jobName (e): The name assigned to the job
- jobOwner: The local login name on the submitting host of the user who submitted the batch job
- gridjobOwner (u) : The Global user Distinguished Name
- resourcesUsedWalltime: Maximum amount of real time (wall-clock elapsed time) which the job needs to execute.
- Queue (q): The name of the queue in which the job currently resides
- ctime: The time that the job was created
- execHost: If the job is running, this is set to the name of the host or hosts on which the job is executing
- execHostTreated (n): this is set to the number of host or hosts on which the job is executing

After solving all semantic data problems, the data was transformed merging in a single entry the job behaviour in a given status. For that reason the data was summarized using the maximum aggregate function for resourcesUsedWalltime. The rest of the fields have the same value across the aggregated functions.
It is important to mention that the data generation, collection, and processing was not adding any distortion to the load of the target Grid environment. The data is by default generated by the Grid middleware and job manager of each site and the collection and processing were done in a set of dedicated offsite servers.

### 4.2 Dynamic templates applied to the NGS workload logs

In order to analyze how dynamic templates perform in a production Grid environment, the Greedy algorithm was implemented together with the predictor generator as explained in [9] using the *Mean* predictor in all templates for the full set of past data points. Also user-supplied data was not incorporated into any Workload log, only real information was considered.
The algorithm was run using the NGS workload logs (Table 3) producing as a result a walltime prediction per job. The output is summarized in Table 4 showing each template and prediction errors ordered by decreasing level of accuracy. The table contains:

- A first column with the list of characteristics that composes each template.
- A second column with the average walltime in seconds taken by all jobs belonging to each template.
- A third column printing the percentage of mis-prediction.
- And finally, the number of jobs that have been fitted into each template.

The experiment exhibits that the use of dynamic templates in this production Grid environment is not encouraging predicting a walltime future submission with an average error of 47 percent in almost 50 percent of the cases and a shocking average of 106 percent of error for the rest of the examples.
It is important to notice that the most relevant jobs characteristics, such as the job names and parameters, are in most of the cases not taken into account and that there are two templates, (u) and (u-gn) that fit an important number of jobs. A preliminary conclusion may say that

most of the time a user in the NGS submits the same type of jobs to the same Grid node. But the real reason can be found mining the workload logs: 46% of the jobs have names or identifications assigned by default by the Grid middleware, such as 'STDIN' and 14% of the jobs are either the same executable using slightly different names or the same name for different executables.
Therefore, given that an important set of characteristics (such as, job identification, parameters) are not normalized (as it is expected in a typical Grid environment) the outcome of the dynamic template allocation process is distorted producing that jobs are grouped together by an inadequate set of parameters, such as the user and the grid node.

| Template | Average WallTime | Average Error | % Error | # |
|---|---|---|---|---|
| (q-n) | 295.95 | 59.85 | 20.22 | 7 |
| (e-u-q) | 36.62 | 11.03 | 30.13 | 94 |
| (u-gn) | 94.81 | 30.92 | 32.62 | 2783 |
| (u-n) | 180.96 | 82.45 | 45.56 | 188 |
| (u-q) | 166.96 | 88.47 | 52.99 | 502 |
| (u) | 311.07 | 178.12 | 57.26 | 4444 |
| (e) | 452.85 | 271.48 | 59.95 | 984 |
| (e-u-gn) | 65.87 | 41.10 | 62.39 | 253 |
| (u-gn-n) | 16.75 | 10.66 | 63.65 | 22 |
| (e-u-n) | 42.35 | 29.52 | 69.71 | 4 |
| (e-gn-q-n) | 157.25 | 118.38 | 75.28 | 1 |
| (gn-n) | 374.94 | 308.87 | 82.38 | 97 |
| (e-q) | 303.60 | 252.53 | 83.18 | 195 |
| (e-n) | 319.35 | 266.70 | 83.51 | 271 |
| (n) | 1289.13 | 1130.78 | 87.72 | 120 |
| (e-gn) | 635.45 | 558.01 | 87.81 | 392 |
| (gn-q) | 1186.14 | 1076.28 | 90.74 | 53 |
| (gn) | 1399.95 | 1292.69 | 92.34 | 74 |
| (empty) | 1464.39 | 1363.45 | 93.11 | 199 |
| (e-gn-q) | 187.88 | 225.04 | 119.78 | 383 |
| (q) | 708.68 | 884.09 | 124.75 | 386 |
| (e-gn-n) | 238.19 | 303.82 | 127.55 | 129 |
| (u-gn-q) | 19.27 | 25.64 | 133.04 | 191 |
| (e-u) | 30.77 | 41.97 | 136.39 | 2093 |

Table 4 Dynamic tempates in the NGS

The initial conclusion of this experiment is that a set of restrictions related to the quality of workloads data needs to be considered at the time of applying dynamic templates. But, even applying those restrictions, it is not clear that the use of the walltime mean with the smallest confident of all possible templates may produce the best prediction.

Therefore, in the next Chapter a new definition of job similarity using selective templates is presented and implemented. The results are validates against the dynamic template method using a restricted NGS workload data.

## 5. Weighted templates.

### 5.1 Weighted template algorithm

One of the main problems found in the experiment carried out in Chapter 4 is that when a relevant set of job characteristics are not normalized the results are inaccurate. Reading it from a different angle, the experiment showed that there are some characteristics which are more important than others and that they need to be consider at any attempt to forecast a job performance. For that reason, it is important to define a method that creates templates that take into account the idea of having characteristics with different weights. Characteristics heavier than others have higher priority and have to be mandatory for job performance estimations. Another important aspect to be considered is the prediction accuracy that certain templates are offering to the Grid environment. Post-mortem examination of all possible templates can be used to weigh templates.
Therefore, the Weighted Template method, instead of using all available characteristics without preference as it is done in the dynamic template algorithm, divides them into two sets: a first set is compose of the job name (e), the Grid user (u) and the list of parameters (p) leaving a second set with any other available characteristics.
The algorithm comprises two faces, a prediction face that creates templates and calculates the possible future execution time and a post-evaluation face that incorporates the prediction accuracy level.

**Prediction face**: Given a new job submission

1. Divide the job characteristics into two sets as described before.
2. For each possible combination of templates compose of characteristics from the first set (Excluding the empty template)
    a. Select from historical information the level of accuracy for each template
    b. If a template does not have any accuracy level, include it with level 0
3. From all possible templates from point 2, select all templates with the highest prediction accuracy.

a. If the selection produces only one template
   i. Calculate the Mean of the walltime
b. Otherwise, for each selected template
   i. Extend them using all possible combination of characteristics belonging to the second set and apply the dynamic template algorithm (reference) to all of them.
   ii. Select the template with smallest confident interval and calculate the Mean of the walltime

4. Otherwise, select the mean of the template with highest prediction accuracy as a prediction result.

**Incorporation of prediction accuracy face**: Once the job complete execution

1. Select the closest mean from all calculated templates and increase the prediction accuracy by one.

| Template (First/ Second Group | Average WallTime | Average Error | % Error | # |
|---|---|---|---|---|
| (e-u) (gn-q) | 350.20 | 69.19 | 19.76 | 333 |
| (e)(q-n) | 344.32 | 75.15 | 21.83 | 21 |
| (e-u-p) (gn-q-n) | 150.96 | 34.47 | 22.84 | 210 |
| (e-u-p)(gn) | 58.95 | 13.47 | 22.86 | 13 |
| (e-u) (empty) | 180.02 | 47.58 | 26.43 | 2 |
| (e-u-p)(q-n) | 582.52 | 155.55 | 26.70 | 1 |
| (e-u) (gn-q-n) | 468.66 | 145.04 | 30.95 | 396 |
| (e-u)(gn-n) | 1274.27 | 432.81 | 33.97 | 39 |
| (e-u-p) (gn-q) | 217.60 | 82.47 | 37.90 | 118 |
| (e-u-p) (gn-n) | 573.88 | 230.83 | 40.22 | 13 |
| (e-u)(gn) | 162.61 | 65.64 | 40.37 | 85 |
| (e-u-p) (empty) | 353.88 | 157.96 | 44.64 | 4 |
| (e)(n) | 1759.43 | 871.07 | 49.51 | 12 |
| (e)(gn-n) | 1980.61 | 1105.68 | 55.83 | 2 |
| (e)(gn-q-n) | 1872.13 | 1098.25 | 58.66 | 5 |
| (e)(empty) | 231.49 | 276.76 | 119.56 | 7 |
| (e)(gn) | 252.65 | 397.37 | 157.28 | 8 |

Table 5 Weighted templates

The algorithm considers that in Grid environments there are key characteristics that have to be treated with different priority and also weights those templates that perform better than others.

### 5.2 Weighted Template in the NGS

In order to validate the new method, a subset of the workload data was prepared. All anomalies were cleaned leaving only records with clear identification of job name, parameters, and Grid user.

Table 5 shows the results of applying the Weighted Template algorithm to the filtered NGS workload data. It can be seen that the template column contains a pair of values: the first component corresponds to the template defined using the first set of characteristics and the second one correspond to the template defined using the second set of characteristics.

Also the dynamic template method was applied to the same workload obtaining the results shown in Table 6.

The results show that the Weighted Template algorithm have an average error of 29 percent when predicting job submissions and that the dynamic templates algorithm have a prediction error of 64 percent. As a result the Weighted Template algorithm produces results 54 percent more accurate.

| Template | Average WallTime | Average Error | % Error | # |
|---|---|---|---|---|
| (e) | 379.66 | 172.16 | 45.35 | 455 |
| (e-q) | 518.26 | 242.19 | 46.73 | 22 |
| (q) | 494.33 | 232.87 | 47.11 | 294 |
| (u-n) | 992.80 | 559.08 | 56.31 | 20 |
| (e-n) | 953.08 | 607.72 | 63.76 | 8 |
| (e-gn-q) | 2868.55 | 1976.60 | 68.91 | 1 |
| (gn) | 2453.74 | 2000.61 | 81.53 | 20 |
| (empty) | 2374.72 | 1998.73 | 84.17 | 6 |
| (u-gn) | 107.29 | 91.53 | 85.30 | 22 |
| (n) | 79.48 | 74.41 | 93.62 | 363 |
| (gn-q) | 1629.29 | 1612.98 | 99.00 | 3 |
| (u) | 297.03 | 320.63 | 107.95 | 37 |
| (u-q) | 484.74 | 593.19 | 122.37 | 15 |
| (u-gn-q) | 294.52 | 605.42 | 205.56 | 2 |
| (e-gn) | 891.95 | 1974.73 | 221.40 | 1 |

Table 6 Dynamic template

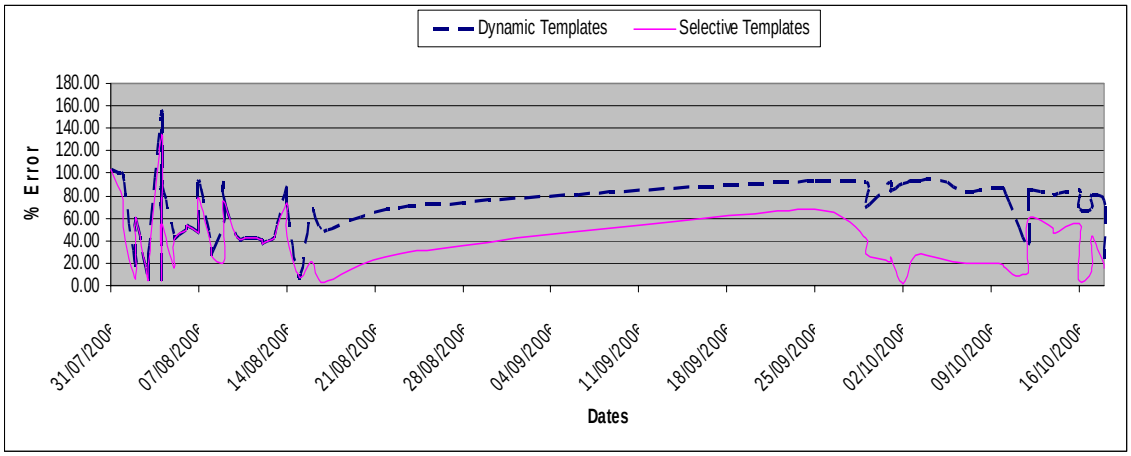


Figure 1 Dynamic templates vs. Selective templates for a specific NGS job (DNALDH)

In order to understand and explain how both algorithms are behaving and where the improvements are Figure 1 shows the error percentage of both algorithms for a given job (in this example called DNALDH).
In this figure it can be seen that when the historical information has not yet classified the templates with different prediction accuracy level, both algorithms are presenting a similar error level. But, there is a point in the timeline where the learning curve pays off and, while the dynamic template method still uses the normal distribution with the smallest confident, the Weighted Templates method starts using the weighted information.

## 6. Conclusions.

A key component in making a more accurate prediction in future job submissions is the characterization of jobs and the definition of similarity. The main idea is to group jobs together in order to use previous information to forecast future submissions. In this paper the best solution that uses dynamic templates is selected and tested in a production Grid environment, the NGS. It is also evaluated where it is possible to improve this solution using a different approach that takes into account Grid features.

This research shows than defining similarity using two set of characteristics, a binding first level that concentrate only the relevant Grid characteristics that uses a prediction technique compose of a weight function based on historical prediction accuracy and a dynamic second level that uses the reminder of the characteristics using a prediction technique based on statistical methods achieves in average a prediction errors than is 54 percent lower than those achieved by only using dynamic templates.

There are still some pending issues to be considered in this new approach. Historical data can be composed of a large dataset and have different weight depending on its age. Therefore, improvements in amount of data and ageing-related queries need to be done. Also, during this experiment the Mean of each template has been calculated to forecast jobs walltime. A different set of data mining and statistical tools needs to be tested and compared to understand where some techniques are more convenient than others

## 7. Acknowledgements.

This research has been partially supported by the CoreGRID European Network of Excellence (FP6-004265)

## 8. References.

[1] Ralf Gruber, Pieter Volgers, Alessandro De Vita, Massimiliano Stengel, Trach-Minh Tran: Parameterisation to tailor commodity clusters to applications. Future Generation Comp. Syst. 19(1): 111-120 (2003)
[2] Ralf Gruber, Vincent Keller, Pierre Kuonen, Marie-Christine Sawley, Basile Schaeli, Ali Tolou, Marc Torruella, Trach-Minh Tran: Towards an Intelligent Grid Scheduling System. PPAM 2005: 751-757
[3] PA Dinda and P. A. The statistical properties of host load. Scientific Programming, 1999.
[4] Peter A. Dinda, , and David R. O Hallaron. Host load prediction using linear models.

Cluster Computing, Volume 3 and Number 4:265 - 280, December 2000

[5] Yuanyuan Zhang, Wei Sun, , and Yasushi Inoguchi. Cpu load predictions on the computational grid. Cluster and Grid computing, 2006

[6] Rosa M. Badia, Jesús Labarta, Judit Gimenez,Francesc Escale Workshop on Grid Applications and Programming Tools (GGF8), 2003. DIMEMAS: Predicting MPI applications behavior in Grid environments.

[7] Jesús Labarta, Sergi Girona, Toni Cortés. Analyzing scheduling policies using Dimemas. 3rd Workshop on environment and tools for parallel scientific computation. Parallel Computing, vol. 23 (1997) pp 23-34. Ed. Elsevier.

[8] PACE home page http://www.cimosa.de/IctSupport/PACE5.html

[9] Warren Smith, Valerie E. Taylor, and Ian T. Foster. Using run-time predictions to estimate queue wait times and improve scheduler performance. Proceedings of the job Scheduling Strategies for Parallel Processing, Lecture Notes In Computer Science; Vol. 1659:202 - 219, 1999.

[10] Warren Smith, Ian Foster, and Valerie Taylor. Predicting Application Run Times Using historical information.

[11] K. Kurowski, A. Oleksiak, J. Nabrzyski, A. Kwiecien, M. Wojtkiewicz, M. Dyczkowski, F. Guim, J. Corbalan, J. Labarta: Multi-criteria Grid Resource Management using Performance Prediction Techniques. Integrated Research in Grid Computing, Springer, to appear (presented at CoreGrid Integration Workshop, Pisa, November 2005)

[12] Hui Li, Juan Chen, Ying Tao, David Groep and LexWolters. Improving a local learning technique for queue wait time predictions. Cluster and Grid computing, 2006.

[13] A. Goyeneche, F. Guim, I. Rodero, G. Terstyansky, J. Corbalan, "Extracting performance hints for grid users using Data mining techniques: a case study in the NGS", The Mediterranean Journal of Computers and Networks, Vol. 3, No. 2, 2007, p 52-61

[14] Allen Downey. Predicting queue times on space-sharing parallel computers. In international parallel processing symposium, 1997.

[15] Richard Gibbons. A historical application profiler for use by parallel schedulers. job Scheduling Strategies for Parallel Processing 1997, 1997

[16] The UK National Grid Service Site: www.ngs.ac.uk

[17] Globus toolkit Site: http://www.globus.org/toolkit/

[18] PBS Pro job Manager Site: www.openpbs.org